Untitled

Jesse Elder

2024-01-22

```r
library(ggfortify)
```

```
## Loading required package: ggplot2
```

```r
#library(compositions)
library(stats)
library(compositions)
```

```
## Welcome to compositions, a package for compositional data analysis.
## Find an intro with "? compositions"

##
## Attaching package: 'compositions'

## The following objects are masked from 'package:stats':
##
##     anova, cor, cov, dist, var

## The following object is masked from 'package:graphics':
##
##     segments

## The following objects are masked from 'package:base':
##
##     %*%, norm, scale, scale.default
```

```r
library(DESeq2)
```

```
## Loading required package: S4Vectors

## Loading required package: stats4

## Loading required package: BiocGenerics

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:compositions':
##
##     normalize, var

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, aperm, append, as.data.frame, basename, cbind,
##     colnames, dirname, do.call, duplicated, eval, evalq, Filter, Find,
##     get, grep, grepl, intersect, is.unsorted, lapply, Map, mapply,
##     match, mget, order, paste, pmax, pmax.int, pmin, pmin.int,
##     Position, rank, rbind, Reduce, rownames, sapply, setdiff, sort,
##     table, tapply, union, unique, unsplit, which.max, which.min

##
## Attaching package: 'S4Vectors'

## The following objects are masked from 'package:base':
##
##     expand.grid, I, unname

## Loading required package: IRanges

## Loading required package: GenomicRanges

## Loading required package: GenomeInfoDb

## Loading required package: SummarizedExperiment

## Loading required package: MatrixGenerics

## Loading required package: matrixStats

##
## Attaching package: 'MatrixGenerics'

## The following objects are masked from 'package:matrixStats':
##
##     colAlls, colAnyNAs, colAnys, colAvgsPerRowSet, colCollapse,
##     colCounts, colCummaxs, colCummins, colCumprods, colCumsums,
##     colDiffs, colIQRDiffs, colIQRs, colLogSumExps, colMadDiffs,
##     colMads, colMaxs, colMeans2, colMedians, colMins, colOrderStats,
##     colProds, colQuantiles, colRanges, colRanks, colSdDiffs, colSds,
##     colSums2, colTabulates, colVarDiffs, colVars, colWeightedMads,
##     colWeightedMeans, colWeightedMedians, colWeightedSds,
##     colWeightedVars, rowAlls, rowAnyNAs, rowAnys, rowAvgsPerColSet,
##     rowCollapse, rowCounts, rowCummaxs, rowCummins, rowCumprods,
##     rowCumsums, rowDiffs, rowIQRDiffs, rowIQRs, rowLogSumExps,
##     rowMadDiffs, rowMads, rowMaxs, rowMeans2, rowMedians, rowMins,
##     rowOrderStats, rowProds, rowQuantiles, rowRanges, rowRanks,
##     rowSdDiffs, rowSds, rowSums2, rowTabulates, rowVarDiffs, rowVars,
##     rowWeightedMads, rowWeightedMeans, rowWeightedMedians,
##     rowWeightedSds, rowWeightedVars

## Loading required package: Biobase

## Welcome to Bioconductor
##
##     Vignettes contain introductory material; view with
##     'browseVignettes()'. To cite Bioconductor, see
##     'citation("Biobase")', and for packages 'citation("pkgname")'.

##
## Attaching package: 'Biobase'

## The following object is masked from 'package:MatrixGenerics':
##
##     rowMedians

## The following objects are masked from 'package:matrixStats':
```

```
##
##     anyMissing, rowMedians
```

```
#install.packages("BiocManager")
library(BiocManager)
```

```
## Bioconductor version '3.16' is out-of-date; the current release version '3.18'
##   is available with R version '4.3'; see https://bioconductor.org/install
```

```
#BiocManager::install("DESeq2")
library(ggrepel)
#BiocManager::install("metagenomeSeq")
library(metagenomeSeq)
```

```
## Loading required package: limma

##
## Attaching package: 'limma'

## The following object is masked from 'package:DESeq2':
##
##     plotMA

## The following object is masked from 'package:BiocGenerics':
##
##     plotMA

## Loading required package: glmnet

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following object is masked from 'package:S4Vectors':
##
##     expand

## Loaded glmnet 4.1-8

##
## Attaching package: 'glmnet'

## The following object is masked from 'package:compositions':
##
##     rmult

## Loading required package: RColorBrewer
```

```
#BiocManager::install("ALDEx2")
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x lubridate::%within%()    masks IRanges::%within%()
## x dplyr::collapse()        masks IRanges::collapse()
## x dplyr::combine()         masks Biobase::combine(), BiocGenerics::combine()
## x dplyr::count()           masks matrixStats::count()
## x dplyr::desc()            masks IRanges::desc()
## x tidyr::expand()          masks Matrix::expand(), S4Vectors::expand()
## x dplyr::filter()          masks stats::filter()
## x dplyr::first()           masks S4Vectors::first()
## x dplyr::lag()             masks stats::lag()
## x tidyr::pack()            masks Matrix::pack()
## x BiocGenerics::Position() masks ggplot2::Position(), base::Position()
## x purrr::reduce()          masks GenomicRanges::reduce(), IRanges::reduce()
## x dplyr::rename()          masks S4Vectors::rename()
## x lubridate::second()      masks S4Vectors::second()
## x lubridate::second<-()    masks S4Vectors::second<-()
## x dplyr::slice()           masks IRanges::slice()
## x tidyr::unpack()          masks Matrix::unpack()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
#devtools::install_github("ggloor/ALDEx_bioc")
library(ALDEx2)
```

```
## Loading required package: zCompositions
## Loading required package: MASS
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
##
##     select
##
## Loading required package: NADA
## Loading required package: survival
##
## Attaching package: 'NADA'
##
## The following object is masked from 'package:IRanges':
##
##     cor
##
## The following object is masked from 'package:S4Vectors':
##
##     cor
##
## The following object is masked from 'package:compositions':
##
##     cor
##
## The following object is masked from 'package:stats':
##
##     cor
##
## Loading required package: truncnorm
## Loading required package: lattice
## Loading required package: latticeExtra
##
## Attaching package: 'latticeExtra'
##
```

```
## The following object is masked from 'package:ggplot2':
##
##     layer
```

```
library(ggpubr)
#install.packages("ggplotify")
library(ggplotify)
library(vegan)
```

```
## Loading required package: permute
## This is vegan 2.6-4
```

Load in Data

```
amr<-read.csv("~/Desktop/NoyesLab/OREI_Calfs/Noyes_Project_019_AMR++_Results/SNPconfirmed_AMR_analytic_
metadata<-read.csv("~/Desktop/NoyesLab/OREI_Calfs/sample_data_06_08_2023.csv")
rownames(metadata)<-metadata$SampleId
#dim(amr)

#separate amr classifications into own columns
amr_classification<-amr %>% separate_wider_delim(cols = gene_accession,delim='|',
                                                  names = c("accession","type" ,"class", "mechanism","gr
amr_classification<-as.data.frame(amr_classification)
```

Initial Object Creation

Data Cleaning and Exclusion Function

```
#####
#####BEGIN DATA CLEANING AND EXCLUSION FUNCTION
#####
clean_and_exclude<- function(amr_class,DIMfilter){

  class_cols<-c("accession","type","class","mechanism","group") #classification columns to keep

  ###SNP Confirmation issues
  #Manually fix bugged Rifampin
  amr_class$group[amr_class$accession=="MEG_8674"]<-
    paste0(amr_class$accession[amr_class$accession=="MEG_8674"],"|RequiresSNPConfirmation")

  #Exclude low sensitivity SNP confirmed AMR
  amr_class<-amr_class[!grepl("RequiresSNPConfirmation",amr_class$group),]


  ###Naming schemes
  #Keep only real USDA samples, no controls
  amr_class<-amr_class[,colnames(amr_class) %in% class_cols | grepl("USDA",colnames(amr_class))]

  #Reformat names to match metadata
  colnames(amr_class)<-gsub("(USDA\\d+)_.*",'\\1',colnames(amr_class))
  #sum(colnames(amr_class) %in% metadata$SampleId) == dim(amr_class)[2] #Confirm matching sample_ids
  ###Filter on DIM
  # if (DIMfilter == "after"){  #Days in Milk after birth
  #   new_DIM<-metadata$SampleId[metadata$DIM>0]                #List of IDs
  #   amr_class<-amr_class[,colnames(amr_class) %in% class_cols | colnames(amr_class) %in% new_DIM]  #K
  #
  # } else if(DIMfilter == "before"){ #Days in milk before birth
  #   new_DIM<-metadata$SampleId[metadata$DIM<=0]               #List of IDs
  #   amr_class<-amr_class[,colnames(amr_class) %in% class_cols | colnames(amr_class) %in% new_DIM]  #K
  # } else if(DIMfilter =="all"){}


  ###Exclude 0-read samples
  zerocol<-which(colSums(amr_class[,6:dim(amr_class)[2]])==0) #vector of 0 columns
  amr_class<-amr_class[, -(zerocol+5) ] #Remove 0 columns, shift 5 to match columns
  #Find a way to operationalize this

  return(amr_class)
  #sum(colSums(amr_classification[,6:dim(amr_classification)[2]])==0)
  #View(amr_classification)
  #Check on any 0 counts
  #colnames(amr)[colSums(as.matrix(amr[,2:dim(amr)[2]]))==0]
  #amr_classification[rowSums(amr_classification[,6:dim(amr_classification)[2]])==0,5])

}
```

Count Processing Function

```
#####
#####BEGIN COUNT PROCESSING FUNCTION
#####
class_process<- function(x) {

  ###Group by and sum according to classification of AMR
  #Agg ***sum*** accross classifications is correct
  mechsums<-aggregate(.~mechanism,data=x[,c(4,6:dim(x)[2])],FUN=sum)
  classsums<-aggregate(.~class,data=x[,c(3,6:dim(x)[2])],FUN=sum)
  groupsums<-aggregate(.~group,data=x[,c(5,6:dim(x)[2])],FUN=sum)

  #Convert column to rowname, delete columns
  rownames(mechsums)<-mechsums$mechanism
  rownames(classsums)<-classsums$class
  rownames(groupsums)<-groupsums$group
  mechsums<- mechsums %>% dplyr::select(-c(mechanism))
  classsums<- classsums %>% dplyr::select(-c(class))
  groupsums<- groupsums %>% dplyr::select(-c(group))

  #mechsums<-rowsum(amr_classification[,c(6:dim(amr_classification)[2])],group=amr_classification$mecha
  #classsums<-rowsum(amr_classification[,c(6:dim(amr_classification)[2])],group=amr_classification$clas
  #groupsums<-rowsum(amr_classification[,c(6:dim(amr_classification)[2])],group=amr_classification$grou
  ###Remove 0 counts
  mechsums<-mechsums[rowSums(mechsums)!=0,]
  #sum(rowSums(mechsums)==0) #Fixed it
  classsums<-classsums[rowSums(classsums)!=0,]
  #sum(rowSums(classsums)==0) #Fixed it
  groupsums<-groupsums[rowSums(groupsums)!=0,]
  #sum(rowSums(groupsums)==0) #Fixed it


  #transpose for analysis
  mech_t<-as.data.frame(t(mechsums) )
  class_t<-as.data.frame(t(classsums) )
  group_t<-as.data.frame(t(groupsums) )

  #Put into object
  mat_list<-list(mech_t,class_t,group_t)
  names(mat_list)<-c("mechanism","class","group")

  return(mat_list)

}
```

Generate Dataset

```
#amr_class_clean_pre<-clean_and_exclude(amr_classification,"before")
#amr_class_clean_post<-clean_and_exclude(amr_classification,"after")
amr_class_clean_all<-clean_and_exclude(amr_classification,"all")

amr_list<-class_process(amr_class_clean_all)
```

Grab Metadata

```
#####
#####GRAB FULL METADATA
#####
amr_meta<-amr_list$mechanism #Load in amr_list for join
amr_meta$SampleId<-rownames(amr_meta)

#Left join to grab metadata fields
amr_meta<-left_join(amr_meta,metadata[c("SampleId","CaseOrControl","FarmId","DIM","Batch")],by="SampleI
  dplyr::select("SampleId","CaseOrControl","FarmId","DIM","Batch") %>%  #keep only metadata
  column_to_rownames("SampleId") %>%
  mutate("Calving" = ifelse(DIM>=0,"Postnatal","Prenatal")) #Divide DIM into case-control

#Confirm metadata is in same order, identical rownames regardless of type
#identical(rownames(amr_list$class),rownames(amr_meta))
```

Split Data by Farm for Analysis

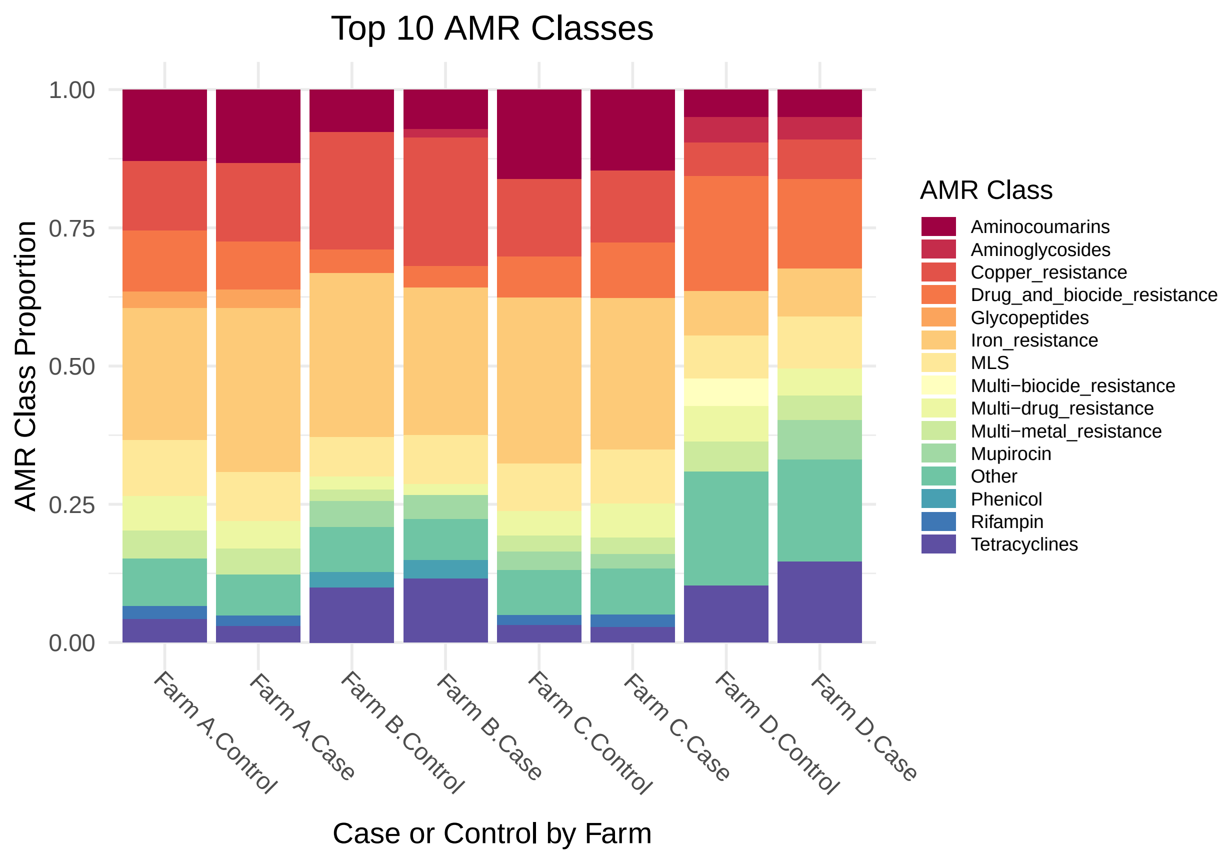
```
#####
#####SPLIT DATA INTO FARMS
#####
amr_farm<-lapply(amr_list,function(x) split(x,amr_meta$FarmId) )    #Split counts
amr_farm_meta<-split(amr_meta,amr_meta$FarmId)   #Split metadata

#Confirm metadata is in same order, identical rownames regardless of type, Farm
#identical(rownames(amr_farm$mechanism$`Farm A`),rownames(amr_farm_meta$'Farm A'))
#identical(rownames(amr_farm$class$`Farm B`),rownames(amr_farm_meta$'Farm B'))
```

Figure Generation and Statistical Analysis

Barplot

```
##################################START BARPLOT SECTION -- WHOLE DATASET
####Maybe start function here, input is FULL class_process object

###Merge transposed dataset with metadata
amr_list$class$SampleId<-rownames(amr_list$class)
class_t_meta<-amr_list$class %>% left_join(metadata[c("SampleId","CaseOrControl","FarmId")],by=c("Sampl
#sum(is.na(class_t_farm$WeeksToInfection)) #Confirmed no unmatched

#colnames(class_t_meta)
class_t_meta<-class_t_meta[,!( colnames(class_t_meta) %in% c("SampleId") )]  #Remove all unwanted metad
#class_t_farm_clr<-class_t_farm_clr[,c(1:48,52,60)]  #Remove all unwanted metadata fields

###Aggregate counts by Case-Control & Farm, *Average
class_agg<-aggregate(.~CaseOrControl + FarmId,data=class_t_meta,FUN = mean)

#Convert feature counts to proportion of reads per sample
class_agg[,!(colnames(class_agg) %in% c("CaseOrControl","FarmId"))]<-t( apply(   #Needs transpose to fi
  class_agg[,!(colnames(class_agg) %in% c("CaseOrControl","FarmId"))],1,
  function(x) x/sum(x,na.rm=TRUE)
) )

#Long format convert all features
class_agg_long<-gather(class_agg,amr,count,Acetate_resistance:Zinc_resistance)



#####Filter down to only the Top 10
#Group dataframe into Farm and CaseOrControl, take Top 10 counts of each
top_10<-class_agg_long %>% group_by(FarmId,CaseOrControl) %>%
  arrange(desc(count),.by_group = T) %>% top_n(10,count)
#View(top_10)

#Add an Other category, proportion of all other classes
others<-cbind( data.frame(amr=rep("Other",8)),
```

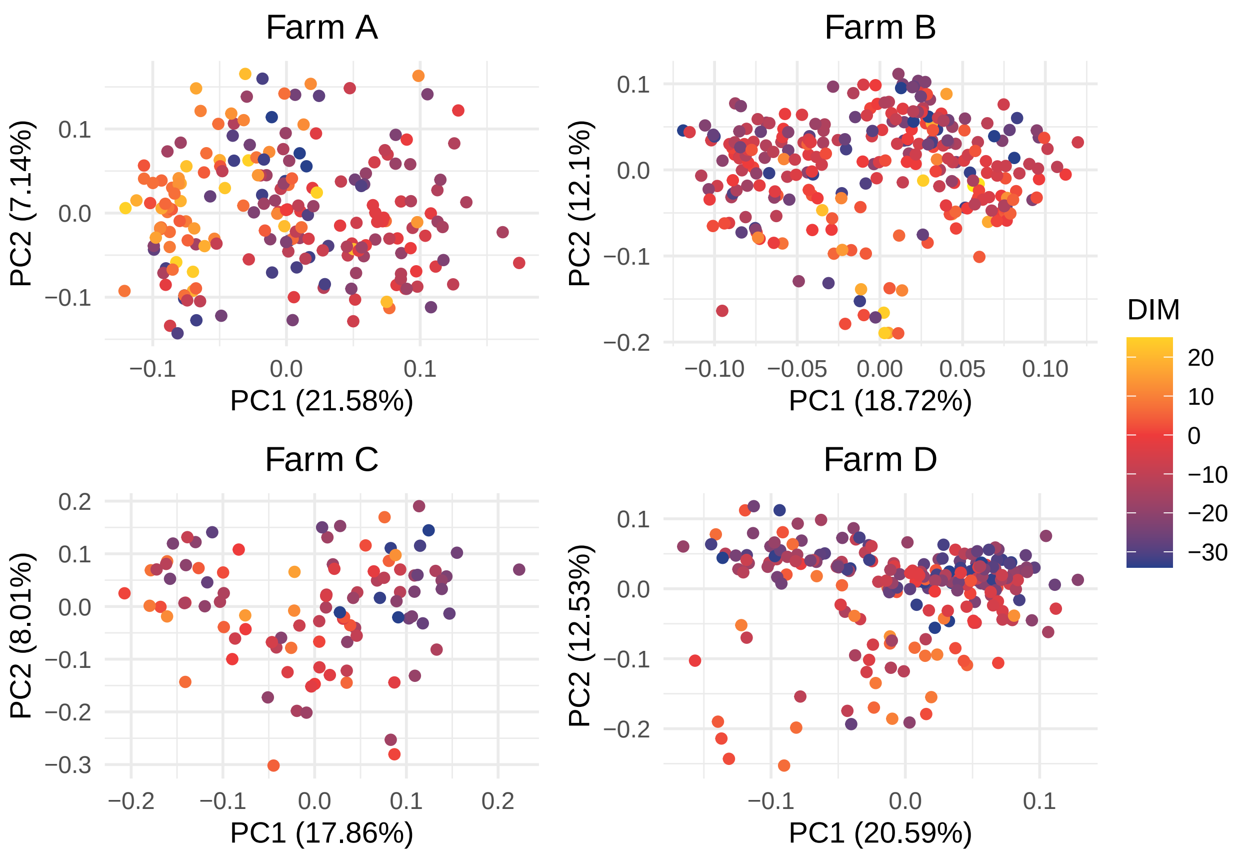
```r
               aggregate(count~CaseOrControl + FarmId,data=top_10,function(x) 1-sum(x)) )
top_10<-rbind(top_10,others)



###Make classic barplot
#Create & reorder interaction variable for x-axis
top_10$FarmCase<-factor(interaction(top_10$FarmId,top_10$CaseOrControl),
                        levels= c("Farm A.Control","Farm A.Case","Farm B.Control","Farm B.Case",
                                  "Farm C.Control","Farm C.Case","Farm D.Control","Farm D.Case")
)

#barplot
ggplot(data=top_10,aes(x=FarmCase,y=count,fill=amr)) +
  geom_bar(stat="identity") +
  theme_minimal() +
  scale_fill_manual(name="AMR Class", values=colorRampPalette(brewer.pal(11,"Spectral"))(15) ) +
  theme(legend.title = element_text(size=10)) +
  theme(axis.text.x = element_text(angle = -45,hjust=0.15,vjust=-0.4)) +
  theme(legend.text = element_text(size=7)) +
  theme(legend.key.height= unit(3, 'mm'),
        legend.key.width= unit(5, 'mm')) +
  ggtitle("Top 10 AMR Classes") +
  theme(plot.title = element_text(hjust=0.5)) +
  xlab("Case or Control by Farm") +
  ylab("AMR Class Proportion")
```



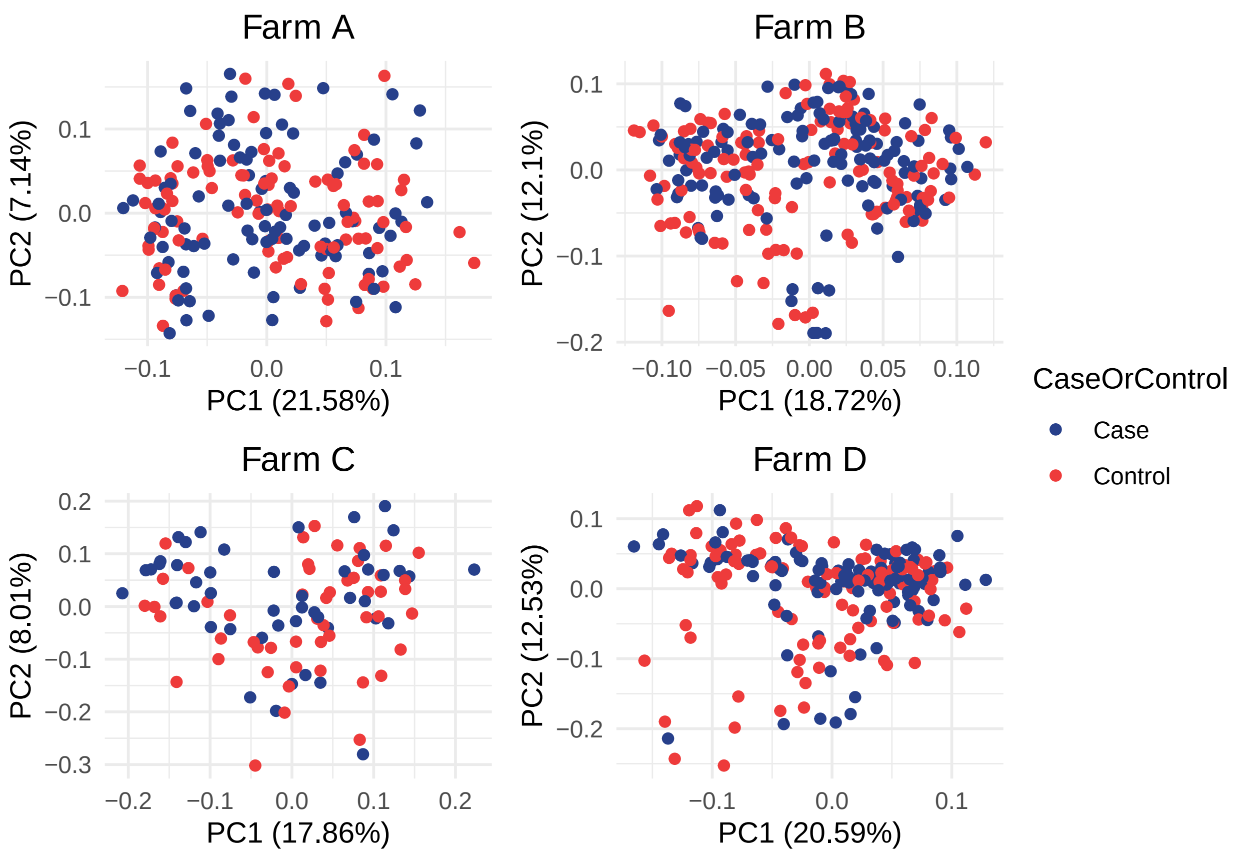
This figure shows the Top 10 classes of AMR present in each grouping of Farm & Case/Control as well as the proportion of all other AMR classes. Iron resistance is consistently in the Top 10 for each group and is the most common class of AMR for all but Farm D. Copper resistance is also common in Farms A, B, & C, most notably in Farm B. The distribution of Farm D is visibly different than the other Farms. Farm D has Drug and biocide resistance and tetracyclines being most common and classes not in the Top 10 making up a larger proportion of the AMR reads in the samples. There are clear differences between Farms, but differences between Staph aureus-positive and negative cows are small and not consistent.

Principal Components Analysis

```
#####
#####START PCA SECTION
#####

###Apply CLR normalization to SPLIT amr object
amr_clr<-lapply(amr_farm$mechanism,function(x) as.data.frame(clr(x)) )

#Apply PCA to clr-transformed data
amr_pca<-lapply(amr_clr,function(x) prcomp(x)) #use only numeric values

A<-autoplot(amr_pca$`Farm A`,data = amr_farm_meta$`Farm A`,colour = "DIM") + scale_color_gradient2(midpo
            #loadings=T,loadings.label=T,loadings.size=0.1,loadings.label.size=2,loadings.color="blue",
B<-autoplot(amr_pca$`Farm B`,data = amr_farm_meta$`Farm B`,colour = "DIM") + scale_color_gradient2(midpo
C<-autoplot(amr_pca$`Farm C`,data = amr_farm_meta$`Farm C`,colour = "DIM") + scale_color_gradient2(midpo
D<-autoplot(amr_pca$`Farm D`,data = amr_farm_meta$`Farm D`,colour = "DIM") + scale_color_gradient2(midpo


ggarrange(A,B,C,D,common.legend = T,legend="right")
```



```
#####Now color by Case or Control

A<-autoplot(amr_pca$`Farm A`,data = amr_farm_meta$`Farm A`,colour = "CaseOrControl") +
  scale_color_manual(values = c("royalblue4","brown2")) +
  theme_minimal() + ggtitle("Farm A") + theme(plot.title = element_text(hjust=0.5))
            #loadings=T,loadings.label=T,loadings.size=0.1,loadings.label.size=2,loadings.color="blue",
B<-autoplot(amr_pca$`Farm B`,data = amr_farm_meta$`Farm B`,colour = "CaseOrControl") +
  scale_color_manual(values = c("royalblue4","brown2")) +
  theme_minimal() + ggtitle("Farm B") + theme(plot.title = element_text(hjust=0.5))
C<-autoplot(amr_pca$`Farm C`,data = amr_farm_meta$`Farm C`,colour = "CaseOrControl") +
  scale_color_manual(values = c("royalblue4","brown2")) +
  theme_minimal() + ggtitle("Farm C") + theme(plot.title = element_text(hjust=0.5))
D<-autoplot(amr_pca$`Farm D`,data = amr_farm_meta$`Farm D`,colour = "CaseOrControl") +
  scale_color_manual(values = c("royalblue4","brown2")) +
  theme_minimal() + ggtitle("Farm D") + theme(plot.title = element_text(hjust=0.5))


ggarrange(A,B,C,D,common.legend = T,legend="right")
```



Principal Components Analysis of AMR mechanisms by Farm and colored by days in milk. 0 days in milk is defined as the day calving occurs with negative and positive values being before and after calving respectively. Each farm shows a divergence in the distribution of AMR present on the cow teat post-calving. When colored by Case or Control, there is no obvious change in AMR composition for the Staph aureus-positive cases.

```
# ###FULL DATASET
# ###Apply CLR normalization to FULL amr object
# amr_clr<-lapply(amr_list,function(x) as.data.frame(clr(x)) )
#
# #Apply PCA to clr-transformed data
# amr_pca<-lapply(amr_clr,function(x) prcomp(x)) #use only numeric values
#
# #Relevant loadings:
# loadingvec<-c("Copper_resistance_protein","Lincosamide_nucleotidyltransferases","Iron_resistance_prot
#                "Drug_and_biocide_SMR_efflux_pumps","Drug_and_biocide_RND_efflux_pumps","Tetracycline_r
#
#
# #PCA plot with loadings
# p<-autoplot(amr_pca$mechanism,data = amr_meta$mechanism,colour = "DIM",
#              loadings=T,loadings.label=T,loadings.size=0.1,loadings.label.size=2,loadings.color="blue"
#   scale_color_gradient2(midpoint=0,low="royalblue4",mid="brown2",high="yellow") + theme_minimal() +
#   facet_wrap(~FarmId)
#
# #Keep only the top 5 loadings
# p$layers[[2]]$data<-p$layers[[2]]$data[loadingvec,]
# p$layers[[3]]$data<-p$layers[[3]]$data[loadingvec,]
# #Change loading width
# p$layers[[2]]$aes_params$size <- 0.2
# p
#
# #Other autoplots sans loadings
# autoplot(amr_pca$class,data = amr_meta$class,colour = "DIM") +
#   scale_color_gradient2(midpoint=0,low="royalblue4",mid="brown2",high="yellow") +
#   facet_wrap(~FarmId)
# autoplot(amr_pca$group,data = amr_meta$group,colour = "DIM") +
#   scale_color_gradient2(midpoint=0,low="royalblue4",mid="brown2",high="yellow") +
#   facet_wrap(~FarmId)
#
#
# #Editors note:
# #Chris used negative binomial model to associate S. aureus vs other taxa
# #Also used the model w/emmeans to calculate *adjusted OTU abundances* How? What does this mean?
# #Came up with adjusted richness/diversity metrics
```

Full PCA (deprecated)

Significance Testing PERMANOVA

```
#######################PERMANOVA
#prcomp used for running PCA
#Aitchison Distance matrix was computed using the vegdist function in 'vegan'
#Aitchison distance matrix (euclidean distance of a centered-log ratio scaled abundance matrix) was com
#test if the differences between time period distance were greater than those within-time distances usi


#SPLIT DATASET
amr_dist<-lapply(amr_farm$mechanism, function(x) vegdist(as.data.frame(clr(x)),method="euclidean") )


amr_test_list<-list()
for(i in names(amr_dist)){
  amr_test<-adonis2(amr_dist[[i]] ~ DIM + CaseOrControl + Batch,data=amr_farm_meta[[i]],method = "aitch
  amr_test_list[[i]]<-amr_test
}

amr_test_list
```

```
## $`Farm A`
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = amr_dist[[i]] ~ DIM + CaseOrControl + Batch, data = amr_farm_meta[[i]], method = "
##                Df SumOfSqs      R2      F Pr(>F)
## DIM             1    188.1 0.02732 5.7229  0.001 ***
## CaseOrControl   1     29.5 0.00428 0.8970  0.557
## Batch           1     28.7 0.00417 0.8743  0.596
## Residual      202   6639.4 0.96423
## Total          205   6885.7 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## $`Farm B`
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = amr_dist[[i]] ~ DIM + CaseOrControl + Batch, data = amr_farm_meta[[i]], method = "a
##                 Df SumOfSqs      R2      F Pr(>F)
## DIM              1    189.9 0.02368 6.7806  0.001 ***
## CaseOrControl    1     47.8 0.00596 1.7061  0.050 *
## Batch            1     23.0 0.00287 0.8212  0.641
## Residual       277   7759.0 0.96749
## Total          280   8019.7 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## $`Farm C`
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = amr_dist[[i]] ~ DIM + CaseOrControl + Batch, data = amr_farm_meta[[i]], method = "a
##                Df SumOfSqs      R2      F Pr(>F)
## DIM             1    172.8 0.03992 4.0927  0.001 ***
## CaseOrControl   1     59.5 0.01375 1.4097  0.101
## Batch           1     43.6 0.01006 1.0319  0.398
## Residual       96   4052.4 0.93627
## Total          99   4328.2 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## $`Farm D`
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
##
## adonis2(formula = amr_dist[[i]] ~ DIM + CaseOrControl + Batch, data = amr_farm_meta[[i]], method = "a
##                 Df SumOfSqs      R2      F Pr(>F)
## DIM              1    251.3 0.03112 7.4908  0.001 ***
## CaseOrControl    1     77.6 0.00961 2.3141  0.009 **
## Batch            1     30.2 0.00373 0.8988  0.539
## Residual       230   7715.9 0.95553
## Total          233   8075.0 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
###Conclusion
#Days in Milk is significant for all 4 farms
```

```
#Days in Milk explains 2-4% of the variance in the farms
#CaseOrControl is significant for Farm D ONLY
#No notable batch effects
#deprecated





#FULL DATASET
#amr_dist<-lapply(amr_list, function(x) vegdist(as.data.frame(clr(x)),method="euclidean") )

#amr_test<-lapply(amr_dist,function(x) adonis2(x ~ DIM + CaseOrControl + FarmId + Batch,data=amr_meta$m

#Adonis PERMANOVA test results
#amr_test$mechanism
#amr_test$class
#amr_test$group

#All independent variables are significant
#FarmId explains roughly 8% of variance
#DIM explains roughly 0.5-0.6% of variance
#CaseOrControl explains roughly 0.3% of variance
#Minimal batch effects
```

PERMANOVA provides some interesting results. Every farm shows a significant difference in the centroid of AMR mechanism counts depending on Days in Milk. This indicates a shift in AMR composition in the teat from pre to post-calving that is consistent regardless of which farm samples were collected at. Moreover, confirmed infection with Staph aureus was associated with a significant shift in teat AMR composition for Farm B and Farm D. However, changes in DIM accounted for only 2-4% of the variance in these groups, and Case vs Control accounted for <1% of variance.

ALDEX2 Significance Testing – Stratified by Farm

```
################################START ALDEX2 SECTION

aldex_amr<- function(amr_list,amr,farm) {
  amrdf<-amr_farm[[amr]][[farm]]  #Select data for each classification/farm
  CorC<-amr_farm_meta[[farm]]     #Select metadata for each farm

  #Transpose class dataframe and exclude SampleId column if necessary
  aldex_classdf<-t( amrdf )

  #Run ALDEX2 model
  class.aldex <- aldex(aldex_classdf, CorC$Calving, mc.samples=100, test="t", effect=TRUE,
                     include.sample.summary=FALSE, denom="all",
                     verbose=FALSE, paired.test=T, gamma=NULL)
  class.aldex$feature<-rownames(class.aldex)

  #Plot 3 significance testing graphs
  #Include built-in volcano plot JIC
  volcano<- as.ggplot( function()
                        aldex.plot(class.aldex, type="volcano", test="welch", xlab="Difference",
```

```r
                           ylab="-1(log10(q))",main= 'Volcano plot',called.cex = 1) #ylim = c(0,1.5),xli
              ,scale = 0.9 )

  listy<-list(class.aldex,volcano)
  names(listy)<-c("class.aldex","volcano")

  return(listy)
}


#Create data structures for each farm
farma<-aldex_amr(amr_farm,"mechanism","Farm A")
```

```
## aldex.clr: generating Monte-Carlo instances and clr values

## conditions vector supplied

## operating in serial mode

## Warning in aldex.clr.function(as.data.frame(reads), conds, mc.samples, denom, :
## values are unreliable when estimated with so few MC smps

## computing center with all features

## aldex.ttest: doing t-test

## aldex.effect: calculating effect sizes
```

```r
farmb<-aldex_amr(amr_farm,"mechanism","Farm B")
```

```
## aldex.clr: generating Monte-Carlo instances and clr values

## conditions vector supplied

## operating in serial mode

## Warning in aldex.clr.function(as.data.frame(reads), conds, mc.samples, denom, :
## values are unreliable when estimated with so few MC smps

## computing center with all features

## aldex.ttest: doing t-test

## aldex.effect: calculating effect sizes
```

```r
farmc<-aldex_amr(amr_farm,"mechanism","Farm C")
```

```
## aldex.clr: generating Monte-Carlo instances and clr values

## conditions vector supplied

## operating in serial mode

## Warning in aldex.clr.function(as.data.frame(reads), conds, mc.samples, denom, :
## values are unreliable when estimated with so few MC smps

## computing center with all features

## aldex.ttest: doing t-test

## aldex.effect: calculating effect sizes
```

```r
farmd<-aldex_amr(amr_farm,"mechanism","Farm D")
```

```
## aldex.clr: generating Monte-Carlo instances and clr values
## conditions vector supplied

## operating in serial mode

## Warning in aldex.clr.function(as.data.frame(reads), conds, mc.samples, denom, :
## values are unreliable when estimated with so few MC smps

## computing center with all features

## aldex.ttest: doing t-test

## aldex.effect: calculating effect sizes
```

```
#farma$volcano

aplot<-ggplot(data=farma$class.aldex,aes(x=diff.btw,y=-log10(we.eBH))) +
  geom_point(aes(colour=(diff.btw>=2 | diff.btw<=-2) & we.eBH<0.05 ),shape=1) +
  geom_text_repel(aes(label=ifelse((diff.btw>=2 | diff.btw<=-2) & we.eBH<0.05,as.character(feature),''))
  labs(caption = c("Higher Pre-Calving","Higher Post-Calving") ) +
  geom_hline(yintercept=-log10(0.05),linetype="dashed",colour="darkgrey") +
  geom_vline(xintercept=c(-2,2),linetype="dashed",colour="darkgrey") +
  ggtitle("Farm A",) + xlab("CLR Difference") + ylab("-Log(q)") +
  theme_minimal() + theme(legend.position = "none") + scale_color_manual(values=c("black","red")) + the
  theme(plot.caption = element_text(hjust=c(1, 0),vjust=10,colour = "darkgrey"))
```

```
## Warning: Vectorized input to `element_text()` is not officially supported.
## i Results may be unexpected or may change in future versions of ggplot2.
```

```
bplot<-ggplot(data=farmb$class.aldex,aes(x=diff.btw,y=-log10(we.eBH))) +
  geom_point(aes(colour=(diff.btw>=2 | diff.btw<=-2) & we.eBH<0.05 ),shape=1) +
  geom_text_repel(aes(label=ifelse((diff.btw>=2 | diff.btw<=-2) & we.eBH<0.05,as.character(feature),''))
  labs(caption = c("Higher Pre-Calving","Higher Post-Calving") ) +
  geom_hline(yintercept=-log10(0.05),linetype="dashed",colour="darkgrey") +
  geom_vline(xintercept=c(-2,2),linetype="dashed",colour="darkgrey") +
  ggtitle("Farm B",) + xlab("CLR Difference") + ylab("-Log(q)") +
  theme_minimal() + theme(legend.position = "none") + scale_color_manual(values=c("black","red")) + the
  theme(plot.caption = element_text(hjust=c(1, 0),vjust=10,colour = "darkgrey"))
```

```
## Warning: Vectorized input to `element_text()` is not officially supported.
## i Results may be unexpected or may change in future versions of ggplot2.
```

```
cplot<-ggplot(data=farmc$class.aldex,aes(x=diff.btw,y=-log10(we.eBH))) +
  geom_point(aes(colour=(diff.btw>=2 | diff.btw<=-2) & we.eBH<0.05 ),shape=1) +
  geom_text_repel(aes(label=ifelse((diff.btw>=2 | diff.btw<=-2) & we.eBH<0.05,as.character(feature),''))
  labs(caption = c("Higher Pre-Calving","Higher Post-Calving") ) +
  geom_hline(yintercept=-log10(0.05),linetype="dashed",colour="darkgrey") +
  geom_vline(xintercept=c(-2,2),linetype="dashed",colour="darkgrey") +
  ggtitle("Farm C",) + xlab("CLR Difference") + ylab("-Log(q)") +
  theme_minimal() + theme(legend.position = "none") + scale_color_manual(values=c("black","red")) + the
  theme(plot.caption = element_text(hjust=c(1, 0),vjust=10,colour = "darkgrey"))
```

```
## Warning: Vectorized input to `element_text()` is not officially supported.
## i Results may be unexpected or may change in future versions of ggplot2.
```

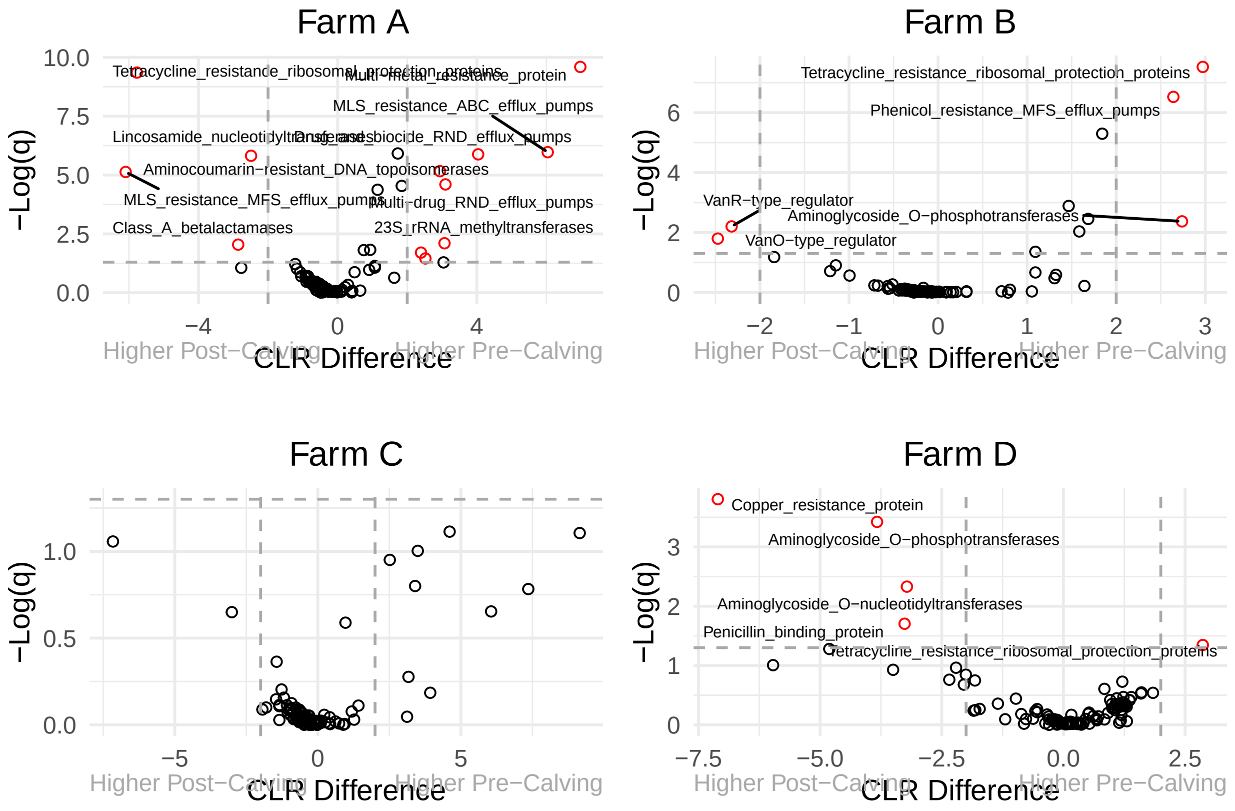
```
dplot<-ggplot(data=farmd$class.aldex,aes(x=diff.btw,y=-log10(we.eBH))) +
  geom_point(aes(colour=(diff.btw>=2 | diff.btw<=-2) & we.eBH<0.05 ),shape=1) +
  geom_text_repel(aes(label=ifelse((diff.btw>=2 | diff.btw<=-2) & we.eBH<0.05,as.character(feature),''))
  labs(caption = c("Higher Pre-Calving","Higher Post-Calving") ) +
  geom_hline(yintercept=-log10(0.05),linetype="dashed",colour="darkgrey") +
```

```r
  geom_vline(xintercept=c(-2,2),linetype="dashed",colour="darkgrey") +
  ggtitle("Farm D",) + xlab("CLR Difference") + ylab("-Log(q)") +
  theme_minimal() + theme(legend.position = "none") + scale_color_manual(values=c("black","red")) + ther
  theme(plot.caption = element_text(hjust=c(1, 0),vjust=10,colour = "darkgrey"))
```

```
## Warning: Vectorized input to `element_text()` is not officially supported.
## i Results may be unexpected or may change in future versions of ggplot2.
```

```r
ggarrange(aplot,bplot,cplot,dplot)
```

```
## Warning: ggrepel: 2 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



```r
#aldex_amr(amr_list$class)
#aldex_amr(amr_list$mechansim)
#aldex_amr(amr_list$group)



########################% of Total Reads
#View(class_agg_long)
```

To the right means normalized abundance of AMR mechanism is higher prior to calving. Farm C shows no individual AMR mechanisms that are differentially abundant before or after calving. Farm B and Farm D both show a higher abundance of Tetracyline resistance (ribosomal protection) prior to giving birth while Farm A shows the opposite trend.

Effect is = diff.btw / max(diff.btw) diff.btw = median difference in clr values between groups (like fold change)

ALDEX2 Significance Testing – FULL DATASET

```
##################################START ALDEX2 SECTION
aldex_amr<- function(amr_list,amr) {

  amrdf<-amr_list[[amr]] #Call data for the type of amr

  ###ALDEX2 Preprocessing/Formatting
  #Make SampleId column for join
  amrdf$SampleId<-rownames(amrdf)

  #Join on metadata to get Case/Control status
  CorC<-amrdf %>% left_join(metadata[c("SampleId","CaseOrControl","DIM")],by=c("SampleId") ) %>%
    mutate("Calving" = ifelse(DIM>=0,"Postnatal","Prenatal")) %>%
    dplyr::select('SampleId','CaseOrControl',"Calving")  #Extract CaseOrControl in correct order for AL

  #CorC

  #Transpose class dataframe and exclude SampleId column
  aldex_classdf<-t( amrdf[-c(which(colnames(amrdf) == "SampleId"))] )


  class.aldex <- aldex(aldex_classdf, CorC$Calving, mc.samples=100, test="t", effect=TRUE,
                       include.sample.summary=FALSE, denom="all", verbose=FALSE, paired.test=T, gamma=NU


  #Plot 3 significance testing graphs
  par(mfrow=c(1,3))
  aldex.plot(class.aldex, type="MA", test="welch", xlab="Log-ratio abundance",
             ylab="Difference", main='Bland-Altman plot',called.cex = 1 )
  aldex.plot(class.aldex, type="MW", test="welch", xlab="Dispersion",
             ylab="Difference", main='Effect plot',called.cex = 1)
  aldex.plot(class.aldex, type="volcano", test="welch", xlab="Difference",
             ylab="-1(log10(q))", main= 'Volcano plot',called.cex = 1) #ylim = c(0,1.5),xlim = c(-2,2)
  #mtext(paste(toupper(amr),"Signficance Plots"), side = 3,line=2, outer = TRUE)     #Add title with am


}

#Make plots and convert to ggplot
p<-as.ggplot( function() aldex_amr(amr_list,"class"),scale = 0.9 )
```

```
## aldex.clr: generating Monte-Carlo instances and clr values

## conditions vector supplied

## operating in serial mode

## Warning in aldex.clr.function(as.data.frame(reads), conds, mc.samples, denom, :
## values are unreliable when estimated with so few MC smps

## computing center with all features

## aldex.ttest: doing t-test

## aldex.effect: calculating effect sizes
```

```r
q<-as.ggplot( function() aldex_amr(amr_list,"mechanism"),scale = 0.9 )
```

```
## aldex.clr: generating Monte-Carlo instances and clr values

## conditions vector supplied

## operating in serial mode

## Warning in aldex.clr.function(as.data.frame(reads), conds, mc.samples, denom, :
## values are unreliable when estimated with so few MC smps

## computing center with all features

## aldex.ttest: doing t-test

## aldex.effect: calculating effect sizes
```

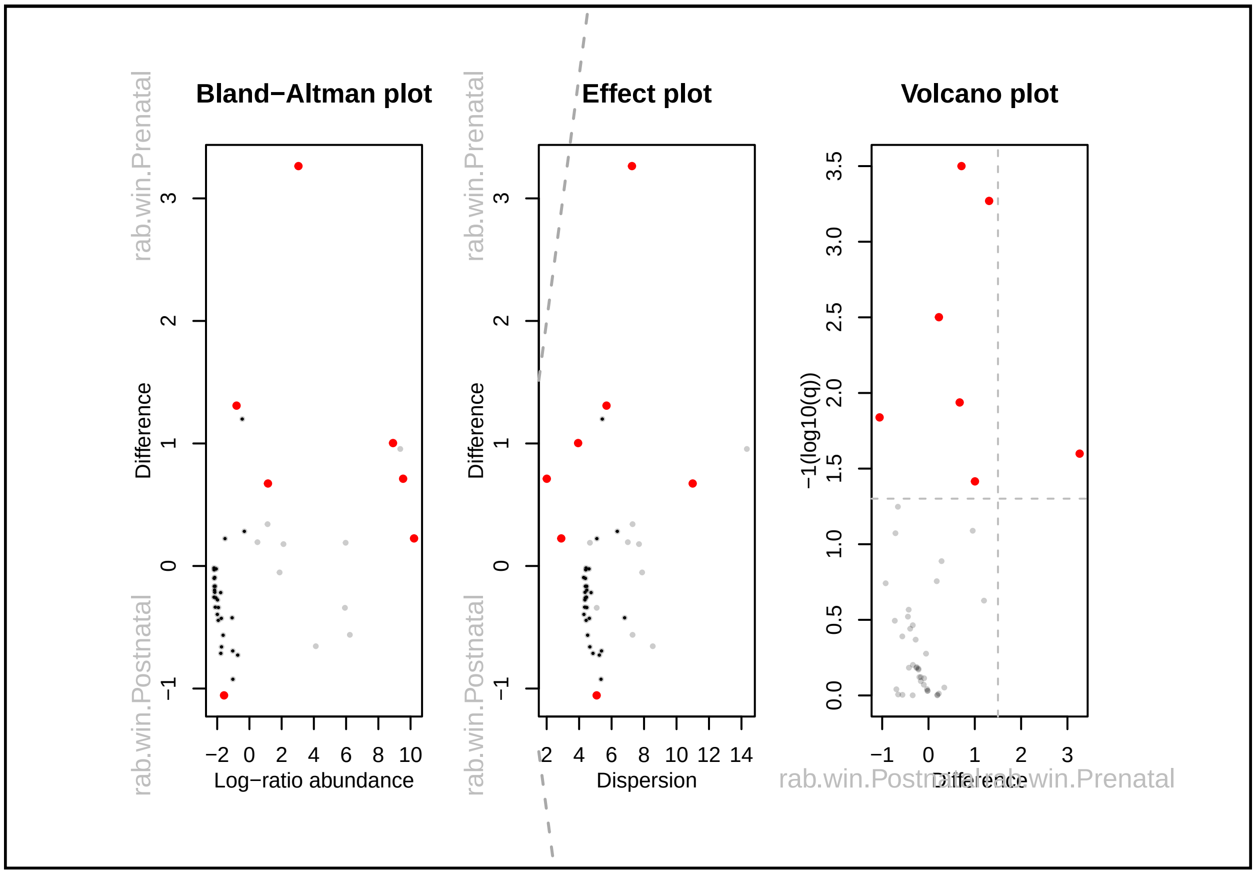
```r
r<-as.ggplot( function() aldex_amr(amr_list,"group"),scale = 0.9 )
```

```
## aldex.clr: generating Monte-Carlo instances and clr values

## conditions vector supplied

## operating in serial mode

## Warning in aldex.clr.function(as.data.frame(reads), conds, mc.samples, denom, :
## values are unreliable when estimated with so few MC smps

## computing center with all features

## aldex.ttest: doing t-test

## aldex.effect: calculating effect sizes
```

```r
#Add borders to plots
p<-p+theme(panel.border = element_rect(colour = "black", fill=NA, size=1) )
```

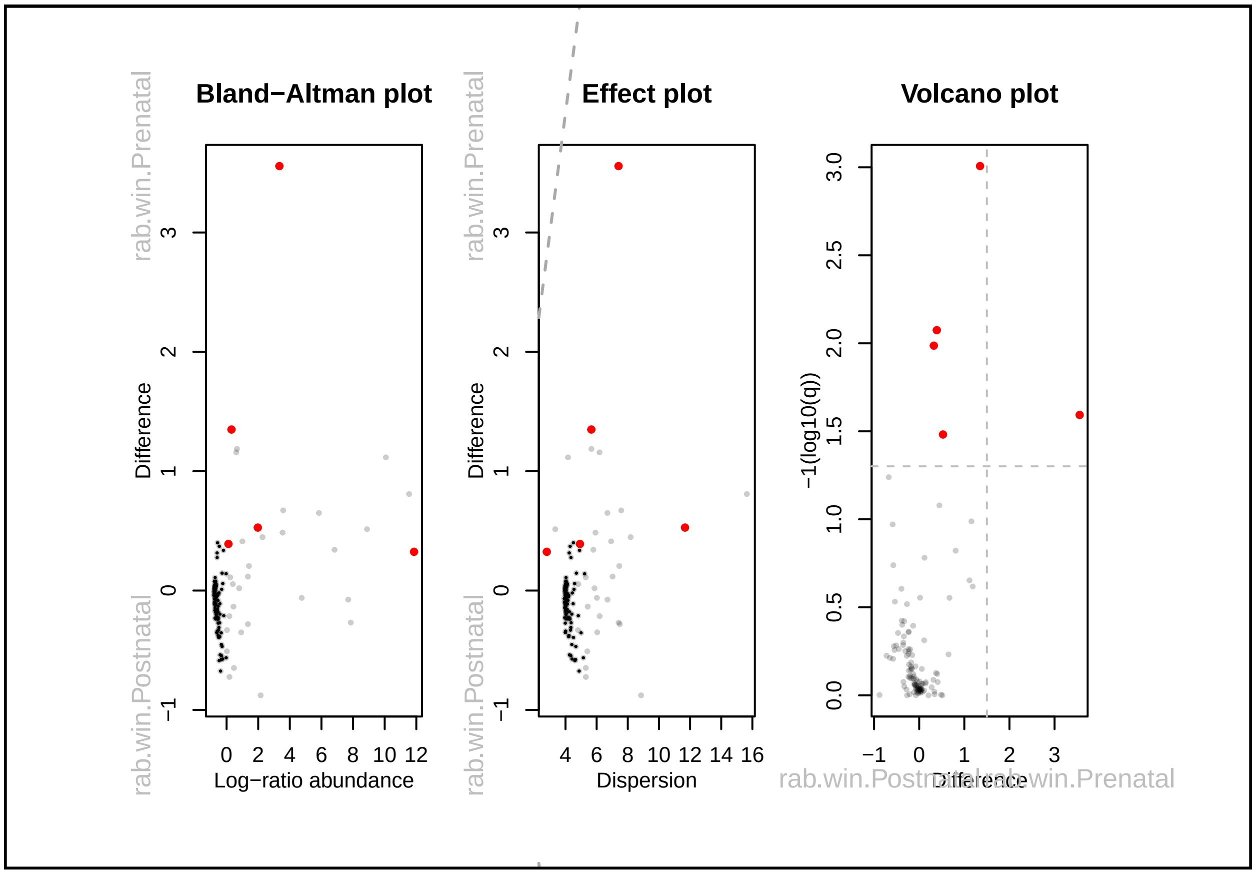
```
## Warning: The `size` argument of `element_rect()` is deprecated as of ggplot2 3.4.0.
## i Please use the `linewidth` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```r
q<-q+theme(panel.border = element_rect(colour = "black", fill=NA, size=1) )
r<-r+theme(panel.border = element_rect(colour = "black", fill=NA, size=1) )
p
```

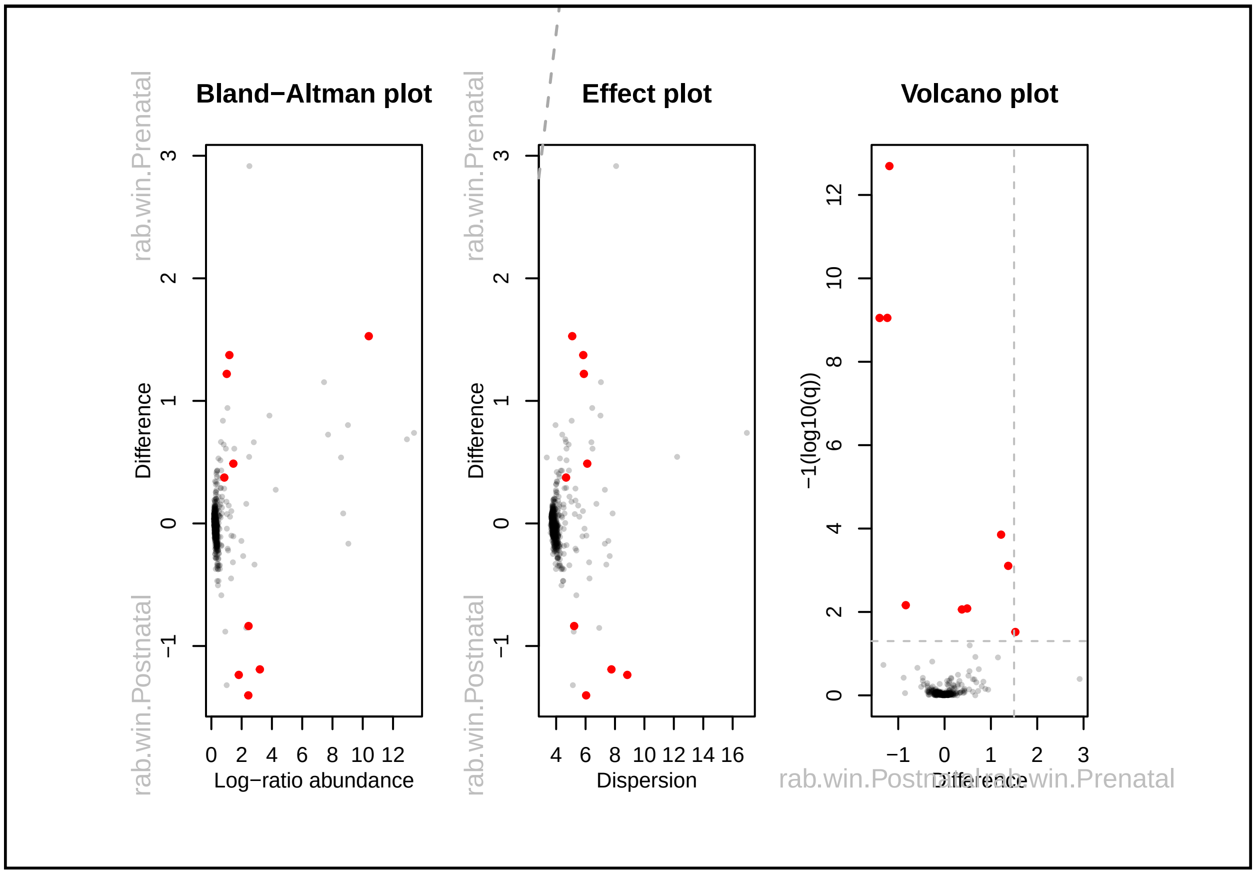


Bland−Altman plot
Effect plot
Volcano plot
rab.win.Prenatal
rab.win.Postnatal
Difference
Log−ratio abundance
Dispersion
−1(log10(q))
rab.win.Postnatal
rab.win.Prenatal

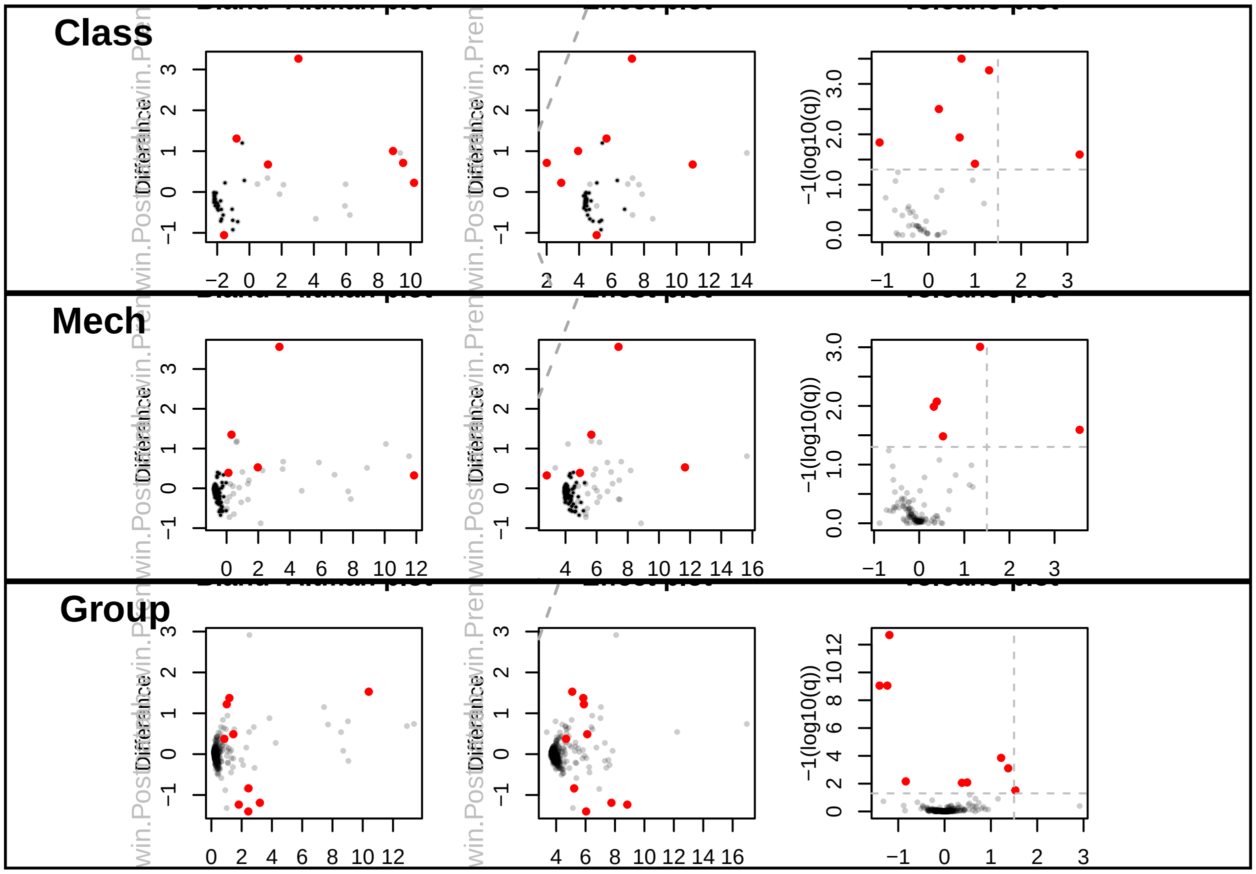
q


Bland−Altman plot
Effect plot
Volcano plot
rab.win.Prenatal
rab.win.Postnatal
Difference
Log−ratio abundance
Dispersion
−1(log10(q))
rab.win.Postnatal
Difference
rab.win.Prenatal

```
r
```



```r
ggarrange(p,q,r,labels=c("Class","Mech","Group"),nrow = 3)
```



```
#aldex_amr(amr_list$class)
#aldex_amr(amr_list$mechansim)
#aldex_amr(amr_list$group)




########################% of Total Reads
#View(class_agg_long)
```